Department of AERONAUTICS and ASTRONAUTICS
STANFORD UNIVERSITY

Final Report

for the Period

April 1, 1990 to March 31, 1993

on

PARTICLE SIMULATION OF HYPERSONIC FLOW

Grant No. AFOSR 90-0232

Submitted to the

Air Force Office of Scientific Research
Building 410, Bolling AFB, D. C. 20332
L. Sakell - Technical Officer

by the

Department of Aeronautics and Astronautics
Stanford University, Stanford, California 94305

Donald Baganoff - Principal Investigator

May 26, 1993

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 26, 1993	Final Technical 4/1/90 to 3/31/93

4. TITLE AND SUBTITLE

Particle Simulation of Hypersonic Flow

5. FUNDING NUMBERS

AFOSR 90-0232

2307/A5

6. AUTHOR(S)

Donald Baganoff

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Department of Aeronautics and Astronautics
Stanford University
Stanford, CA 94305

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
DIRECTORATE OF AEROSPACE SCIENCES
BOLLING AFB, DC 20332-6448

NA

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

AFOSR-90-0232

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION IS UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

A particle code for simulating a rarefied hypersonic flow is developed which accounts for a multiple species gas in thermochemical nonequilibrium and which takes advantage of the parallel architecture of the Intel iPSC/860 supercomputer. Various tests were conducted using a generic blunt body consisting of a hemispherically blunted 60-degree half-angle cone at angle of attack. The different tests conducted show that the performance of the parallel code using 512 nodes exceeds by an order of magnitude the performance obtained for a highly vectorized code run on a single processor of the Cray YMP, that scaleup is found to be nearly linear with the number of processors used, that run time can be greatly reduced by employing a large number of particles, and that runs with 67.5 million particles can be carried out when employing the large memory of the Intel iPSC/860. Other results relate to different applications of the code for studying the near-continuum state of a gas near a cold solid boundary.

14. SUBJECT TERMS

Particle method, rarefied flow, hypersonic flow, direct simulation, Monte Carlo, reaction kinetics, parallel computation

15. NUMBER OF PAGES

38

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	

INTRODUCTION

This final report is for work carried out under Grant No. AFOSR 90-0232 during the three-year period from April 1, 1990 to March 31, 1993.

The principal objective of the research was to further develop Bird's direct simulation Monte Carlo method (DSMC), for modelling rarefied hypersonic flow, so that larger simulations and a wider variety of conditions could be handled than those that had been considered practical in past work. The approach taken was to structure the particle code developed so that it would make effective use of the latest computer technology available, to review all algorithms employed to insure that they were constructed to operate in an efficient way, and to give special attention to studying the relevant physical processes present, in different rarefied flows, so that the modelling employed in the code would be an effective representation for the physical problem being treated in each case.

The computational facilities needed to carry out the work were made available as a result of our close collaboration with members of the Aerothermodynamics Branch at NASA-Ames Research Center, in particular G. S. Deiwert and W. J. Feiereisen. In the 3-year period of the grant this gave our students access to the use of supercomputers such as the Cray-2, Cray Y-MP, Intel iPSC/860 Gamma, Intel iPSC/860 Delta, and the CM-2. More recent updates consisting of the Intel Paragon XP/S-15 and the Cray Y-MP C90 are also being made available. In early stages of their work, the students would either make use of an HP 9000/730 workstation in our laboratory or access the supercomputers from microcomputers in our laboratory. In later stages of their work, they would conduct their research on workstations at NASA-Ames.

The work carried out in this period was closely tied to earlier contributions of several individuals who participated in developing the version of the code used in these studies. Immediately preceding the start of this investigation, Jeffrey D. McDonald had finished a multiple species version of a particle code which he developed to take advantage of the vector capabilities of the Cray family of supercomputers (Cray-2, Cray Y-MP), as well as to take advantage of certain algorithmic improvements introduced by our group. The most recent version of this code is called **PSim2** and is reported in his Ph.D. thesis [1] which dealt with its development. Leonardo Dagum was concurrently working on a similar code which he wrote for the Connection

Machine (CM-2), to take advantage of certain features of that machine [2]. Brian L. Haas was simultaneously devoting his efforts to the development of chemistry models, suitable for use in a simulation employing particles, and he finished his thesis [3] on this subject on December 1990. In this same period, Michael S. Woronowicz was studying rarefied supersonic flows past flat plates and wedges using the original single-species version of the code called **PSim1**, and his efforts were very useful in validating the original coding. His thesis work [4] was finished on June 1991.

After finishing his thesis McDonald turned his attention to the use of the multiprocessor Intel iPSC/860 Touchstone Gamma prototype computer and focused his efforts on developing the new version of his code for it. These efforts led to the creation of **PSim3**, which was later found to contain programming concepts that were not entirely satisfactory in some regards. The knowledge gained from this effort together with large programming segments that could be transferred directly were incorporated by McDonald into a much more successful version called **PSim4**. Haas and McDonald then joined efforts in incorporating chemistry models into this code to create a version that possessed the capabilities we were ultimately seeking. This is the version that has proven to be very useful and exciting and fulfills the expectations and potential one would expect from a supercomputer based on a parallel architecture.

A certain amount of support capability in the form of additional software must be available in order to use complex programs such as **PSim2** and **PSim4** on supercomputers. In our case, one must start with a definition for a three-dimensional body placed in a cubic cartesian grid, which consists of a list of positions and orientations for all the facets making up the body surface. To this one adds the appropriate specification of boundary conditions on the body surface as well as on all the surfaces of the enclosing wind tunnel walls. This geometry generation program (**Geom**) runs on most any computer and creates an output file which is used to initialize the main particle code, **PSim2** or **PSim4**.

The code **PSim2** can be compiled to run on any computer although it was specifically designed to take advantage of the vector capability of the Cray-2 and Cray Y-MP and therefore it runs most quickly on those machines. The code **PSim4** is presently configured to run on both the multiprocessor Intel iPSC/860 Touchstone

Gamma prototype and the Intel iPSC/860 Touchstone Delta computer, although it was designed to be ultimately compatible with a wide variety of parallel architectures.

The output file from either **PSim2** or **PSim4** is post-processed by a control program called **CPlot**. This program allows one to dynamically define new variables in terms of those present in the output file and thus access a wide variety of results. In this way the basic set which is computed becomes a minimal set and one may then expand the set after a simulation is completed. For graphical display, a program **gCPlot** is used which runs on a Silicon Graphics workstation. This program communicates interactively with the **CPlot** program which runs on the computer where the simulation data is stored. When one wants to view the body geometry alone a program **bc2CPlot** is used to convert the output file from **Geom** to a binary file format compatible with both **CPlot** and **gCPlot**.

Each of these support programs was developed by Michael A. Fallavollita, following initial work carried out by McDonald and through further consultation with him. Fallavollita then took on the principal responsibility for further development of the parallel code, conducting an investigation to analyze the performance of the code in the parallel environment represented by the Intel iPSC/860 Touchstone Gamma prototype at NASA-Ames and the iPSC/860 Touchstone Delta at Caltech, and for carrying out the various applications we were interested in investigating. His research led to certain necessary changes in moving the code from the Gamma to the Delta and his thesis work [5], soon to be published, will report on these findings.

For those interested in additional information on the programs described above, at NASA-Ames one may contact

William J. Feiereisen
Phone: (415) 604-4225
E-Mail: feiereis@corvus.arc.nasa.gov

and at Stanford University one may contact

Donald Baganoff
Phone: (415) 723-2849
E-Mail: baganoff@hpsim.stanford.edu

otherwise the following publications can be consulted to gain detailed information.

PUBLICATIONS

A major portion of our reporting has been in the form of publications in archive journals, papers given at conferences, and documents such as a student thesis. A better grasp of the work reported can be obtained if the listing of publications is roughly grouped in terms of subject areas.

Michael S. Woronowicz focused his attention on a study of the flat plate boundary layer and flow past a wedge, specifically for high Mach number rarefied flows. A primary motivation for the work stemmed from the fact that our simulation capability allowed for the use of over 10 million particles and this number was, at the time, two orders of magnitude greater than that used by any other group. Of course, the interest in a large number of particles is related to the fact that it allows one to investigate greater flow detail or study smaller Knudsen numbers. Another motivation for the work was to provide our research group with the experience of thoroughly studying a simple boundary layer so that we could better treat and handle the boundary layers appearing on more complex blunt bodies. The principal publication from Woronowicz' work was his Ph.D. thesis which was also published as a department report.

1. Woronowicz, M.S., "**Application of a Vectorized Particle Simulation to the Study of Plates and Wedges in High-speed Rarefied Flow**," Ph.D. thesis, Stanford University, Stanford, CA, June 1991. This study also appears as a report of the Department of Aeronautics and Astronautics, SUDAAR Report. No. 608, June 1991.

The next two publications resulted from the fact that Woronowicz was able to show that the Knudsen number based on the plate length and the conditions at the plate surface provided a superior correlation for experimental and simulated data than the two well-known, and frequently used, parameters called the viscous interaction parameter and the so-called slip parameter. This discovery resulted from his efforts to obtain the best agreement possible between experimental and simulated results.

2. Woronowicz, M. S. and Baganoff, D., "**Skin Friction and Heat Transfer Correlations for High-Speed Low-Density Flow Past a Flat Plate,**" AIAA Paper No. 91-1314, AIAA 26th Thermophysics Conference, June 24-26, Honolulu, Hawaii, 1991.

3. Woronowicz, M.S. and Baganoff, D., "**Drag and Heat Transfer Correlations for Rarefied Flow Past a Flat Plate,**" Jour. Thermophysics and Heat Transfer, Vol. 7, No. 1, p. 63, January-March 1993.

The following paper was presented at the 18th Rarefied Gas dynamics Symposium, held in Vancouver Canada, July 1992 and will appear in the symposium proceedings for that meeting.

4. Woronowicz, M.S. and Baganoff, D., "**Application of a vectorized Particle Simulation to the Study of a Cold Isothermal Flat Plate in High-Speed Rarefied Flow,**" to be published in the proceedings of the 18th Rarefied Gas Dynamics Symposium, July 1992.

One of Fallavollita's early studies led to an analysis of the computational cost incurred in conducting a particle simulation and he was able to show that an optimum condition exists for carrying out a run. His study showed that, for a given level of statistical uncertainty in the results, the computational cost is minimized if a certain minimum number of particles per cell is used; beyond this minimum, the computational cost is a constant, independent of the number density of particles chosen for the cells. His very preliminary results were reported in a paper given at an APS meeting in 1991.

5. Fallavollita, M.A., Baganoff, D. and McDonald, J.D., "**Computation Cost and Error in Particle Simulation Methods,**" Bulletin of the American Physical Society, Vol. 36, No. 10, p. 2633, Nov. 1991.

His further work on this topic led to a paper that has been accepted for publication in the Journal of Computational Physics. Because this work has yet to appear, it is also presented as an appendix to this report.

6. Fallavollita, M.A., Baganoff, D. and McDonald, J.D., "**Reduction of Cost and Error for Particle Simulations of Rarefied Flows,**" Journal of Computational Physics, to be published.

Fallavollita's principal research efforts were directed towards the application of the Intel iPSC/860 Gamma prototype and the iPSC/860 Delta to the work of our group. His study involved an analysis of code and hardware performance to determine the level achieved and to determine what could be done to improve it still further. One of his early reports was made at a meeting at NASA Ames.

7. Fallavollita, M.A. "**Parallel Implementation and Performance of a Particle Simulation for Modelling Rarefied Hypersonic Flow,**" Computational Aerosciences Conference, NASA-Ames Research Center, August 20, 1992.

A further development of this work led to a presentation at the Symposium on High-Performance Computing for Flight Vehicles, in Arlington, VA on December 20, 1992. This paper appears in the proceeding of the symposium and in the Journal on computing Systems in Engineering.

8. Fallavollita, M.A., McDonald, J.D. and Baganoff, D., "**Parallel Implementation of a Particle Simulation for Modeling Rarefied Gas Dynamic Flow,**" Journal on Computing Systems in Engineering, Vol. 3, Nos 1-4, p. 283, 1992.

Fallavollita has also been interested in using his code to carry out representative runs using a generic blunt body consisting of a cone and a spherical nose. These runs were conducted for various Mach numbers and angles of attack. All of his results will be presented in his Ph.D. thesis [5] which should appear in late summer 1993.

9. Fallavollita, M.A. '**Implementation and Performance of a Particle Simulation Method Suited to MIMD Parallel Computer Architectures,**" Ph.D. thesis, Stanford University, Stanford, CA, August 1993.

As part of his thesis work, Brian L. Haas focused his attention on the modelling of chemical rate processes for a particle method and his research led to the following two publications which appeared in the time frame of the present grant.

10. Haas, B. L. and McDonald, J. D., "**Verification of a Vectorized Particle Method in Simulating Reactive Flows,**" AIAA Paper No. 91-1367, AIAA 26th Thermophysics Conference, June 24-26, Honolulu, Hawaii, 1991.

11. Haas, B. L., "**Models of Energy Exchange Mechanics Applicable to a Particle Simulation of Reactive Flow,**" Jour. Thermophysics and Heat Transfer, Vol. 6, No. 2, p. 200, April-June 1992.

A particle method is ideally suited to handle rarefied flows. However, for certain rarefied flow conditions, the cold gas and resulting higher density in a boundary layer may cause the local conditions near the cold surface to approach the near-continuum state. Therefore, the first place where a simulation, based on a particle method, encounters difficulty is in a cold boundary layer. One of our objectives was to develop a suitable way to handle the flow in a boundary layer without introducing a major disruption in our present coding. Several distinct steps need to be taken in order to accomplish this task. One first needs to find a suitable measure to judge whether the conditions in a cell, in computation space, correspond to rarefied or near-continuum conditions. Once one has a suitable measure, it can then be used to decide whether the code should branch to a special algorithm to handle the near-continuum state.

In order to establish a measure, one needs a theoretical solution for the kinetic relaxation of a gas in a state of gross rest, and this is needed for the general case of a non-Maxwellian gas. In order to obtain a theoretical solution, one must first carry out the mathematics to handle the underlying theory. This has recently been accomplished by Baganoff, and a paper on this subject has been published in the Physics of Fluids A.

12. Baganoff, D., "**Maxwell's Second and Third Order Equations of Transfer for non-Maxwellian Gases,**" Phys. of Fluids A, Vol. 4, No. 1, p. 141, January 1992.

This paper derives the exact moment equations of the Boltzmann equation for molecular models physically more realistic than the Maxwell molecule. These exact equations are highly reduced algebraically because 3 of 5 integrals are shown to be exactly zero; and the 3 that are zero are integrals that are far too complex to integrate. However, as long as one knows they are exactly zero, which is proven in the paper, then their

complexity does not matter. The two remaining integrals are generalizations of the ones that Maxwell himself derived, for the special case of Maxwell molecules, around 100 years ago.

The reason it is so important to know that 3 of 5 integrals are exactly zero, for the more general case, is that these equations are the ones that are used in developing approximate solutions using the moment method, for example, in developing the Burnett terms. If one knows that a certain group is supposed to be exactly zero then it improves any series expansion immensely, because they are automatically eliminated in any analysis when using the highly reduced set, as opposed to using the original set, and they do not survive to complicate later algebra.

A follow-on paper by Baganoff represents an application of the above theory and presents a solution for the case of a gas in a state of gross rest. This is needed to build a theory for handling the near-continuum.

13. Baganoff, D., "**Kinetic Relaxation of a Monatomic Gas in a State of Gross Rest**," Phys. of Fluids A, Vol. 5, No. 5, p. 1260, May 1993.

This is the first time that anyone has constructed an exact theory to handle the case of the hard sphere. The case of the hard sphere is important because it is the one for which a particle simulation is the most securely based, and therefore, its comparison is the most meaningful; and this check has never been made before. This solution is important to us because it describes what happens in a single cell in a single time step in the simulation. It defines for us the proper measure of time so that in the future we can distinguish conditions representing full relaxation from those of partial relaxation, namely, when to switch over from one algorithm to another in our code.

PROGRESS SINCE LAST PUBLICATION

A major portion of Fallavollita's most recent effort has been devoted to the task of making the necessary changes and adjustments to our parallel particle code so that it runs in an efficient way on the Intel iPSC/860 Touchstone Delta at Caltech. This machine is a parallel supercomputer with 528 processors and over 8 gigabytes of memory. The hardware environment represented by the Delta is different in several ways from that for the iPSC/860 Touchstone Gamma prototype at NASA-Ames,

with which we started our work on a parallel code. The principal differences are: a change in the topology of the communication paths between processors, a factor of four greater number of processors, a factor of 8 larger memory, and the absence of a host processor.

The most important element encountered, in making the transition to the larger computer, was the larger number of processors available caused an unanticipated problem that had to be addressed. This can be explained as follows. Our parallel code is based on the division of simulated space into three-dimensional blocks of uniform size, and then each processor on startup is assigned an average of, say, six blocks. As the computation progresses, the amount of time required to process each block is noted and then used to judge the appropriate transfer of blocks (actually the data associated with each block) from heavily loaded processors to more lightly loaded processors,. In this way an attempt is made to more nearly balance the processing time for each processor. However, irrespective of the rules used in selecting which blocks, and how many blocks, are to be transferred, and to which processors they are to be transferred, one must decide on a communication strategy in making the transfer of data. This is where we encountered an unexpected problem.

When the number of processors used is small, in our case less than 128, it makes little difference in the total communication time whether the blocks to be transferred are sent one at a time, in a clearly inefficient way, or whether effort is made to use the communication channels more efficiently by sending multiple blocks simultaneously along separate routes and to separate destinations. When the total communication time is small, one concludes that the appropriate choice is to send the blocks one at a time because the resulting code is far simpler to create, simpler to write, much easier to understand, and as a result far more robust in its use. This is the approach we took and it worked well on the 128-processor iPSC/860 Gamma at NASA-Ames.

However, when attempts were made to employ 256 and then 512 processors on the iPSC/860 Delta at Caltech, an unexpectedly rapid saturation developed and communication started consuming up to 30% of the total computation time. In addition, because of this large cost, proper balance was not being achieved and very large runs were not being carried out successfully. This clearly meant that it was

necessary to reconsider the communication strategy and code used in transferring blocks during load-balancing steps.

The run-time data collected while the details of the problem were being explored are shown in Fig. 1. Shown plotted is the scaleup versus the number of processors used on both the Gamma and the Delta machines. The terminology scaleup, as used here, is the measure of performance obtained by basing results for any number of processors on that for the case of 16 processors, while the problem itself remains unchanged. (This leads to a definition of scaleup where the total computation time per particle for 16 processors is divided by the corresponding time for any other run.) The dashed line gives the ideal speedup which is simply proportional to the number of processors used.

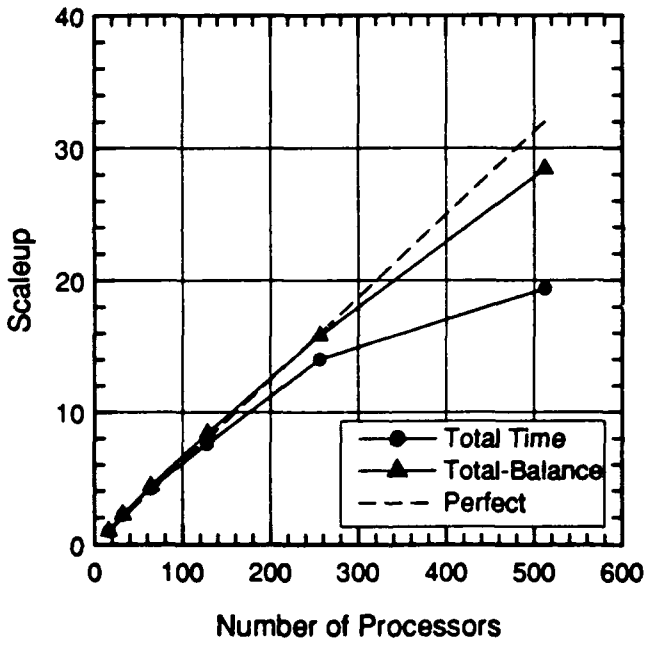

Fig. 1. Scaleup versus the number of processors used on the Gamma and Delta machines. Triangle symbol: total run time including load balancing. Circle symbol: run time for simulation alone, without load balancing.

The triangle symbol represents the measured total computation time which includes load balancing, and the circle symbol represents the measured time for the simulation alone, without load balancing. The difference in time is the time required for load balancing; and it can be seen that roughly 30% of the total run time for 512 processors was consumed by load balancing.

The redesign of the communication strategy carried out by Fallavollita was to attempt to load the communication system to its maximum by determining which blocks could be sent simultaneously, and to do this until all blocks requiring transfer were handled. This is achieved by making sure that no two blocks having the same destination are sent at the same time. This, of course, requires the presence of a great deal of software intelligence; and also leads to a potentially less robust code, requiring careful debugging. His success in accomplishing the task is shown in Fig. 2, where

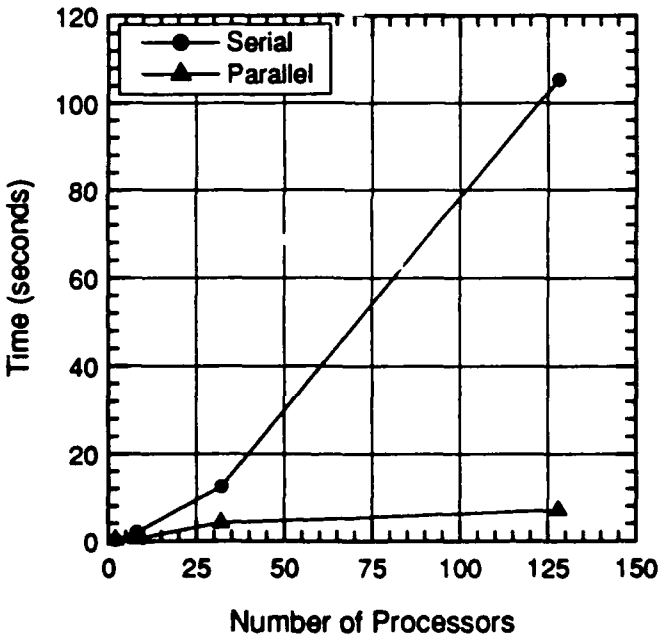

Fig. 2. Run-time for load balancing versus the number of processors used. Circle symbol: serial transmission of blocks. Triangle symbol: simultaneous transmission of bocks having separate destinations.

the run-time for load balancing is displayed versus the number of processors used. The circle symbol represents the original data for serial transmission of blocks and the triangle symbol gives the time found in using his new scheme, where simultaneous transmission is employed for blocks having separate destinations. It is clear from the figure that the new approach reduces the time to a fully acceptable level.

Once the load among the different processors is appropriately balanced and the simulation has reached steady state, the simulation is run for an extended period of time so that time-averaging of appropriate statistical quantities can be carried out, to obtain the desired macroscopic fluid quantities. This period of time can represent the major portion of the total run time for the simulation. At this point one is interested in the performance of the particle code in terms of time per particle per time step and how this measure changes with the number of processors used, see Fig. 3 below.

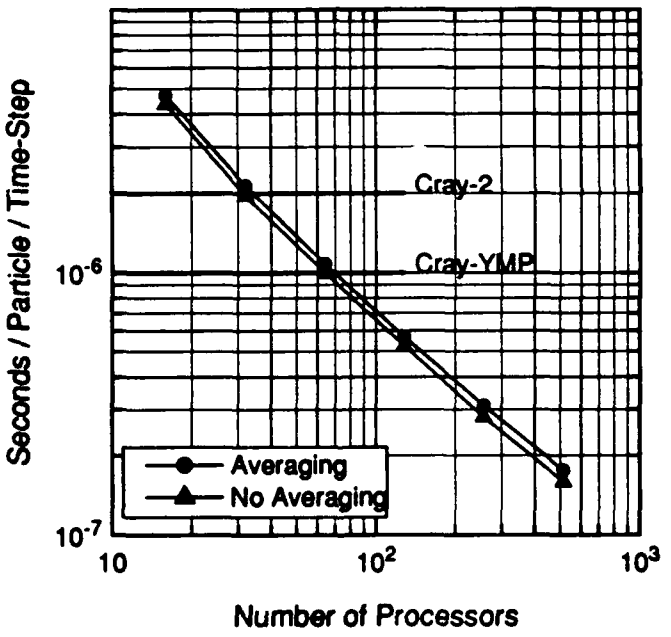

Fig. 3. Performance of the code and hardware after the simulation has reached steady state. Circle symbol: time includes the computation of different averages. Triangle symbol: time does not include the computation of different averages.

This particular measure is meaningful for a particle method because the algorithmic operations used in the method scale in this way. Figure 3 give this information for runs covering both the Gamma and the Delta machines. The difference between the two curves shown represents the amount of time needed to compute and store the data for the averaged values.

The two horizontal lines give the performance of a previously-developed highly-vectorized version of the same code on a single processor on the Cray-2 and the Cray Y-MP. As can be seen, the 528-processor iPSC/860 Touchstone Delta improves the performance by roughly an order of magnitude. The slight curvature seen in the arrangement of data points indicates that the performance is not precisely proportional to the number of processors brought into play, but its deviation over the range shown in comfortably small.

The data of Fig. 3 were collected using an average of 6 blocks-per-processor. This particular number was selected strictly on the basis of an intuitive guess. No information was available to make a better selection. After this part of the study was completed, Fallavollita increased the number of blocks-to-processors by the factor 2^3 to see what effect it would have on a run with 512 processors. To everyone's surprise, the performance improved significantly, and the new value found was 0.11 microseconds/particle/time-step. Clearly, we have much to learn in finding the best operating conditions in using the Intel parallel supercomputer, and Fallavollita will be exploring this question as part of his thesis.

Once the operation with 512 processors was performing well, a large run was carried out to further explore the code's behavior. Much of our testing has been carried out with a generic blunt body consisting of a 60 degree half-angle cone, blunted with a spherical nose, as shown in Fig. 4. The figure presents a contour plot of the temperature field about the body for an angle of attack of 10 degrees, a Mach number of 24, a Knudsen number based on the body diameter of 0.04, and nitrogen gas at a free stream temperature of 210 K. As a first run, the gas composition was limited to nitrogen molecules and atoms and vibrational excitation, dissociation and recombination, but no ionization or radiation. The size of the simulated windtunnel in cell units was 208 tall by 96 deep by 80 long, for a total of 1.6 million cells, and the diameter of the cone was 69 cells. The number of particles used in the simulation

was 67.5 million giving an average of 42.3 particles per cell. The number of cells per block was 4^3 and therefore the corresponding windtunnel dimensions in blocks was 52 by 24 by 20 for an average of 48 blocks per processor. The computation time for both the transient development of the flow and the time averaging period totaled 50 minutes. This is by far the best performance that we have been able to obtain.

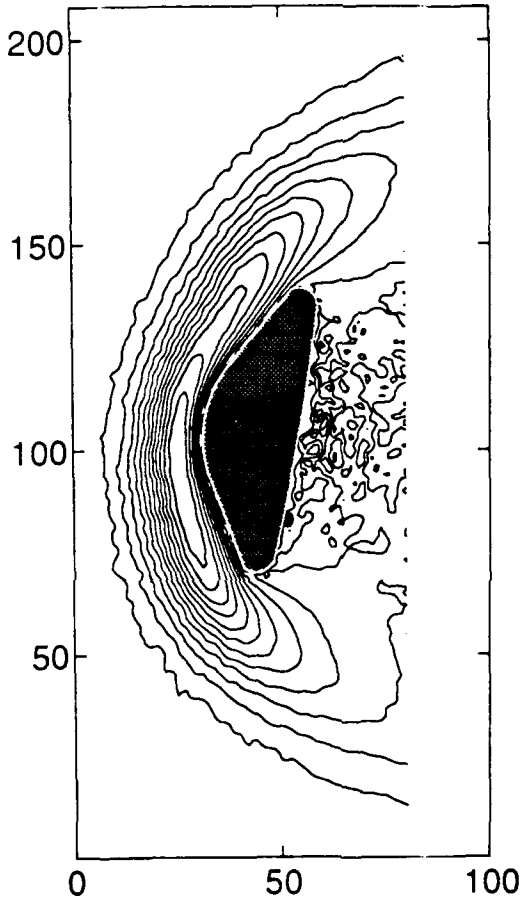

Fig. 4. Temperature distribution in the central plane of the blunt body test discussed in the text.

A more quantitative study of the flow can be pursued by considering specific cuts through the flow field as shown in Fig. 5. The axial cut is chosen to pass as close to the stagnation point as can be determined from an inspection of the data. In view

of the angle of attack of the body, the vertical cut was positioned to show both faces of the cone, and one would expect to see a degree of asymmetry in the data for this cut.

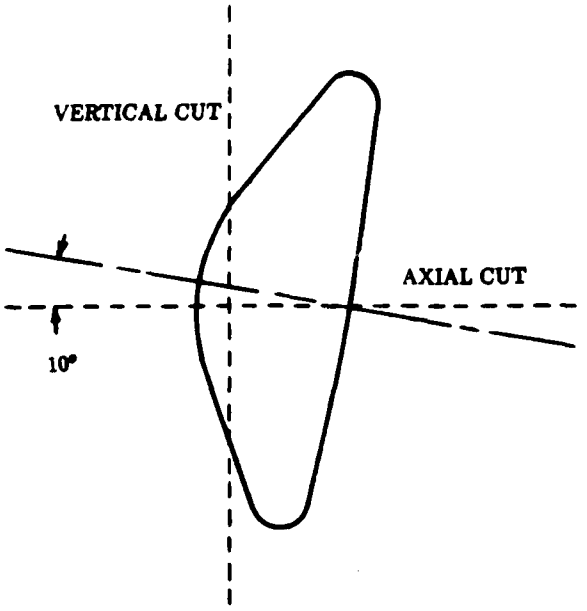

Fig. 5. Positioning of two cuts for displaying results in the following two figures.

Figure 6 shows the temperature and number density for an axial cut located at the position y = 103 cells. The nose of the body can be seen in the data at the position of 33 cells. It is quite evident that the temperature field extends much further ahead of the body than does the number density field. In fact the number density field shows that a large fraction of the 67.5 million particles used in the simulation are found lying close to the body surface, and in this case the number density at the body surface is 33 times the free stream density. The run was carried out for a body temperature of 1,500 K, or approximately 7.5 times free stream temperature, and had it been set to free stream temperature, the number density at the body surface would have been even larger.

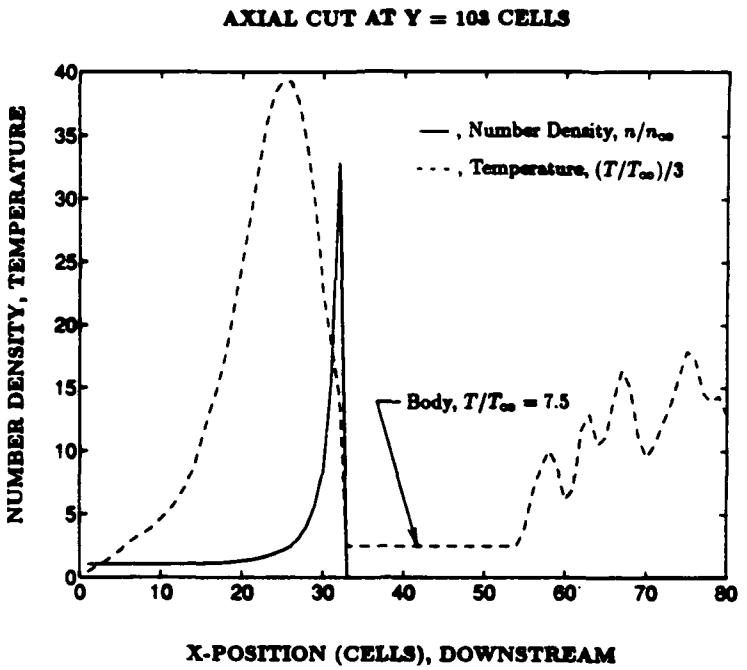

Fig. 6. Temperature and number density distributions along an axial cut passing nearly through the stagnation point.

The transverse cut discussed above is shown in Fig. 7. The bottom or 'compression' side of the body is shown on the left side of the figure and the top or 'expansion' side of the body is shown on the right. The temperature profile does not show relative peaks in the same order as the density profile because of the position of the cut and the angle of attack of the body. This can be seen more clearly by an inspection of Figs. 4 and 5. Here again, it is quite evident that the number density field seems to hug the body while the temperature field extends some distance about the body. This behavior is characteristic of a high Mach number, rarefied flow about a blunt body and where the body temperature is reasonably low. The effect is not so pronounced if the body is slender or if the Mach number is lower. For this case it is clear that the flow is in a near-continuum state near the body surface and that the particles are not being used efficiently in modelling the flow. The particles are needed in the outer flow, where the Navier-Stokes equations clearly do not apply, but the particles are collecting near the body surface where it may be feasible, in fact, to use the Navier-Stokes equations. This situation has prompted our study of the

near continuum, as discussed above, in the hope that the particle method could be interfaced with a near-continuum method so that all particles used in a simulation would remain confined to the region of the flow where they are truly needed.

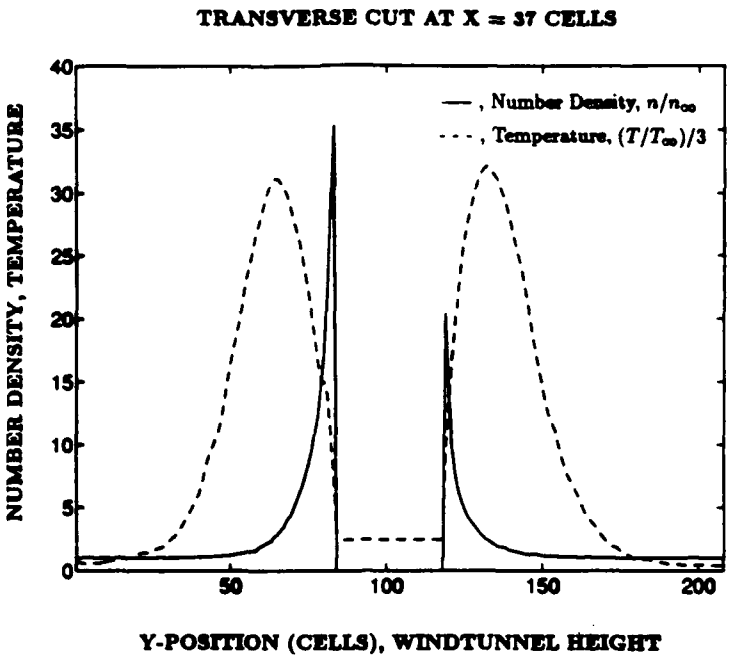

Fig. 7. Temperature and number density distributions along a transverse cut through the body as shown in Fig. 5.

As part of our investigation of the near-continuum and how it should be interfaced with a particle method, we have turned to a study of Couette flow to gain some of the needed insight. For the case of Maxwell molecules, a suitable theoretical solution for Couette flow has been known for 30 years, which is the well-known solution given by Liu and Lees. However, a Maxwell molecule is quite inadequate for modelling a real molecule; a more realistic model is an inverse power molecule, a variable hard sphere molecule, or even a hard sphere molecule. The same analytical work reported by Baganoff and listed as item 12 in the publications discussed above has made possible the generalization of the work by Liu and Lees to these more general molecular models. Terry Denery has carried out the detailed development of

the new theory and his results for the hard sphere molecule are shown in Fig. 8 for the normalized shear stress. The Mach number across the layer was set at 2/3 and the ratio of temperatures (hot/cold) to 3/2. Besides showing that the new theory and simulations agree perfectly in the region where the particle method is expected to hold, the figure also clearly shows where the particle method begins to fail in the near-continuum, namely, in the Navier-Stokes limit, and that the point of failure depends on cell size. As expected, smaller cells allow the particle method to be used further into the near-continuum. The purpose of the theory is to allow one to define the conditions more precisely.

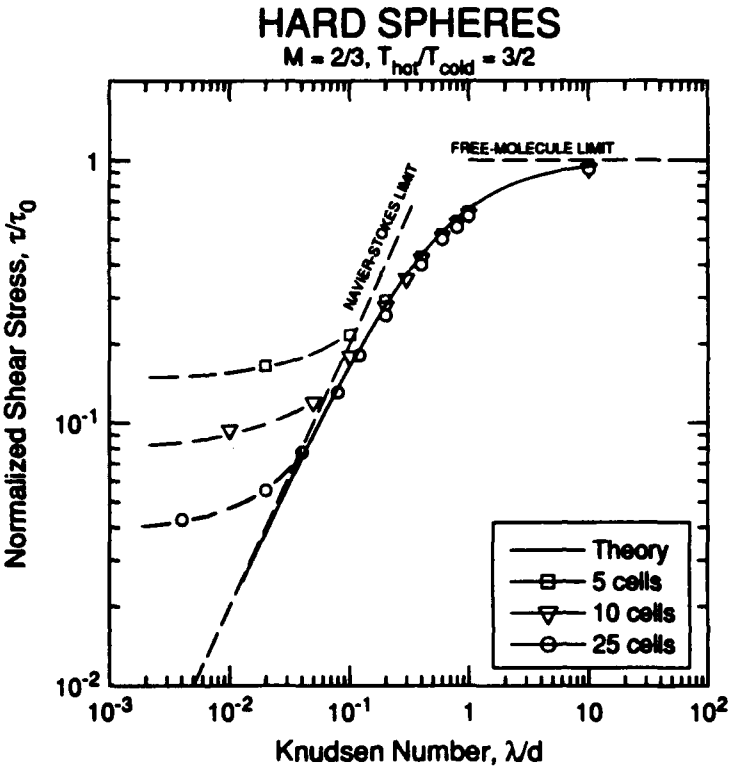

Fig. 8. Normalized shear stress versus Knudsen number for Couette flow assuming a hard-sphere molecular model.

A similar plot for the normalized heat flux is given in Fig. 9 for the same flow conditions. The data from the two figures were used by Denery to develop a suitable

criterion that would allow one to predict where the particle method would fail in a boundary layer. At the present time, it appears that the proper criterion is based on the cell Knudsen number, i.e., the local mean free path length divided by the linear dimension of the cell. If the cell Knudsen number is greater than 0.5 then the particle method is valid, but if the cell Knudsen number is less than 0.5 then the Navier-Stokes equations, or any equivalent continuum set, must be used. This is both a very important result for our work and it would be a fairly easy criterion to implement in a particle code.

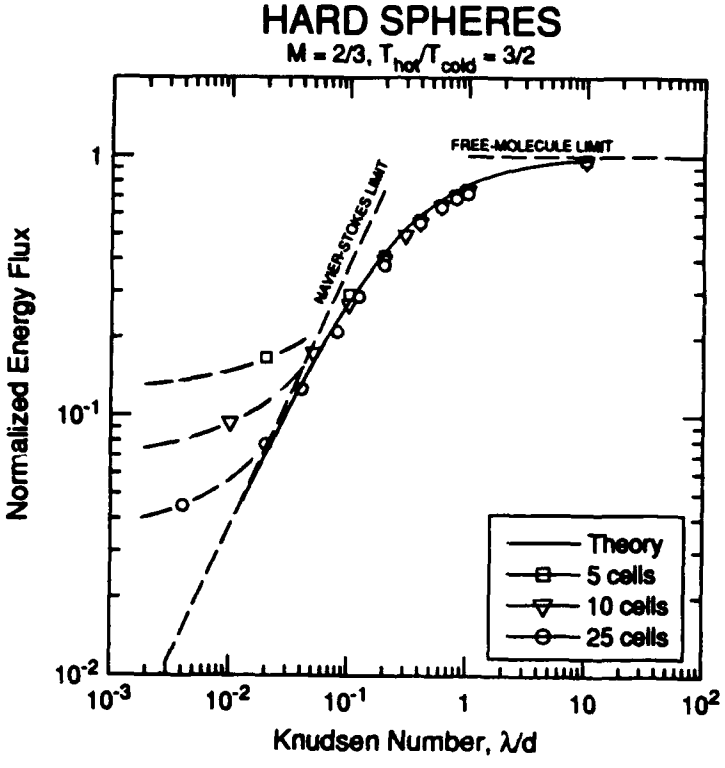

Fig. 9. Normalized heat flux versus Knudsen number for Couette flow assuming a hard-sphere molecular model.

The data in Figs. 8 and 9 were obtained for the single case of $M = 2/3$ and $T_{hot}/T_{cold} = 3/2$. Clearly, a large number of additional cases, especially for large

temperature ratios, should be checked before a final criterion is selected. Once this is done and a suitable near-continuum method is chosen, along with an appropriate method for interfacing it with the particle method, then the theory that led to Figs. 8 and 9 can again be used as a means of checking the resultant hybrid method. That is, to see if the hybrid method gives the correct prediction throughout the entire flow. Even though the pressure is constant across the layer in Couette flow, it is easy to change the local Knudsen number by changing the local temperature and consequently the local density. Therefore, if the temperature ratio across the layer is set to 10 or higher and the flow is rarefied near the hot side, then it may be in the near-continuum near the cold side where the density is greater by a factor of 10 or more.

STUDENTS SUPPORTED

A total of five graduate students were directly supported, at various levels, in the three-year period of the grant. A significant portion of Woronowicz' work was supported by the grant and he finished his Ph.D. thesis on June 1991, after which he accepted a position at NASA Langley pursuing very similar work. Fallavollita, a more recent addition, has been principally supported by the grant and he is expected to finish and submit his Ph.D. thesis by August 1993. Douglas Dahlby, who holds an NSF Fellowship, was supported for a summer period prior to the start of his fellowship and he is presently pursuing his Ph.D. studies while working in our group. Theodore Delianides provided some programming and hardware assistance during an academic year in which we were upgrading our laboratory and while he was finishing a Ph.D. thesis on an allied topic; and Tawei Lou recently joined the research group to pursue a Ph.D. program.

The five Ph.D. theses presented in the following list of references represent either direct sequential investigations or closely connected ones which relate intimately to our overall research objective. Except for the first in the list, each has either directly or indirectly been supported by the present grant.

REFERENCES

1. McDonald, J. D., "A Computationally Efficient Particle Simulation Method Suited to vector Computer Architectures," Ph.D. Thesis, Stanford University, December 1989. Also available as a report of the Department of Aeronautics and Astronautics, SUDAAR Report No. 589, December 1989.

2. Dagum, L., "On the Suitability of the Connection Machine for Direct Particle Simulation," Ph.D. Thesis, Stanford University, June 1990. Also available as a report of the Department of Aeronautics and Astronautics, SUDAAR Report No. 593, June 1990.

3. Haas, B. L., "Thermochemistry Models Applicable to a Vectorized Particle Simulation," Ph.D. Thesis, Stanford University, December 1990. Also available as a report of the Department of Aeronautics and Astronautics, SUDAAR Report No. 600, December 1990.

4. Woronowicz, M.S., "Application of a Vectorized Particle Simulation to the Study of Plates and Wedges in High-speed Rarefied Flow," Ph.D. thesis, Stanford University, Stanford, CA, June 1991. Also available as a report of the Department of Aeronautics and Astronautics, SUDAAR Report No. 608, June 1991.

5. Fallavollita, M.A. 'Implementation and Performance of a Particle Simulation Method Suited to MIMD Parallel Computer Architectures," Ph.D. thesis, Stanford University, Stanford, CA, August 1993 - to be published.

APPENDIX

Reduction of Cost and Error for Particle Simulations of Rarefied Flows

M. A. Fallavollita, D. Baganoff and J. D. McDonald
Department of Aeronautics and Astronautics
Stanford University, Stanford, CA 94305

Introduction

When using a large collection of simulated particles to model a molecular flow on a computer, one is interested in knowing how many particles are needed in each small volume of space in order to properly model the relevant flow physics; and does the use of this number have a beneficial effect on computational cost? These important questions arise irrespective of the particular method used in the simulation, whether it is the method of molecular dynamics (Alder and Wainwright [1]), an approach used in simulating ionized gas motion (Hockney and Eastwood [2]), or the direct simulation Monte Carlo (DSMC) method employed in the study of rarefied gas flows (Bird [3]). In addition, it is clear that the number required to obtain a given level of simulation accuracy when dealing with a nonsteady flow is far larger than that needed for a steady flow where time averaging may be permitted. Our discussion will focus on steady flows and the use of time averaging. In addition, Bird's DSMC method was selected for the study because the calculational effort grows roughly in proportion to the total number of particles N, for which the analysis to be presented is more straightforward, as opposed to N^2 or $NlogN$.

Beyond the requirement of steady flow, the use of time averaging to reduce the effect of statistical fluctuations, which are inherent in a particle method, is based on two assumptions: first, that a sufficient number of particles is in fact present in a computational cell to adequately model the physics of interest; and second, that a time average can be used to replace the cell average obtained if a still larger number of particles were used, or if a large number of repeated runs for the same conditions were carried out. If this replacement is permitted, then the concept of an ensemble average applies to this situation and its exchange with the time average leads to the assumption that the ergodic hypothesis holds. Use of the time average requires that random processes associated with a single cell in space at one time are statistically independent of those associated with the same cell at a different time. In other words, the time interval between samples is greater than the correlation time for the random quantity being averaged. If all these assumptions are valid, then one is able to employ a cell sample size given by $S_c = N_c T$, where N_c is the average number of particles in a single cell and T is the number of time steps used in the time averaging. Because the relative statistical error for an averaged quantity, defined by the ratio $rms/mean$, decreases as $S_c^{-1/2}$ for a statistically independent random process, one

concludes that doubling T allows one to halve N_c, provided that N_c is initially large enough. Therefore, as long as the computational effort is simply proportional to S_c, which is true for Bird's DSMC method, then the simulation with the smaller N_c would be preferred because a smaller demand is placed on the amount of computer memory required, while the computational cost and modelling precision remain the same. Our objective is to fully explore these issues with regard to the DSMC method.

The act of doubling T and halving N_c is certainly limited because one would quickly arrive at a point where too few particles would be present in a cell to adequately model a physical gas flow; an obvious example is $N_c = 1$. Long before this point is reached, it is clear that the simulation would loose efficiency and longer and longer time averages would be required to attempt to obtain the same level of relative statistical error, thus driving up the computational cost. Because no theory exists to guide one in determining how many particles are needed to adequately model given fluid mechanical processes, to evaluate the corresponding computational cost, or to determine whether cost can be reduced by following a particular mode of operation, we conducted a series of numerical experiments using a modified version of the DSMC method to explore these questions. The basic approach followed was to repeatedly run the same simulation for a given problem while varying the duration of the time average and the total number of particles used in the simulation, and then collect the appropriate data to evaluate the level of statistical uncertainty present in the results.

Simulations

The representative problem chosen for study consisted of a two-dimensional rarefied flow past a flat plate placed normal to the oncoming stream as depicted in Figs. 1-3. The free stream Mach number was set at 8 and the Knudsen number, based on the plate height, was fixed at 0.1 to clearly place the flow in the transition regime. A unique characteristic of a rarefied gas flow is that the temperature field extends much further ahead of a blunt body than the density or pressure fields, which can be clearly seen by comparing Figs. 1-3. In order to limit the number of variables in this first study, the simulated gas chosen consisted of diatomic Nitrogen with rotational nonequilibrium (collision number set to 5) but no vibrational nonequilibrium. The molecular collision cross section was modeled using Bird's variable hard-sphere model

destinations.

[4], where the value of the exponent in the inverse power force law was set to 10. The boundary condition on the flat plate was chosen as isothermal, with the plate temperature set equal to 7.5 times the free stream temperature, a value typical for high speed flight in the upper atmosphere. For particles contacting the plate diffuse reflection was assumed. Our intention was to study a fairly straightforward rarefied flow so that the major effects of interest could be easily identified.

In order to properly explore the questions raised, one must have access to a method of simulation that has a very large dynamic range, otherwise the search for modelling limits would be thwarted by the limitations of the simulation itself. In addition to a large dynamic range, the simulation must be computationally efficient because very large simulations as well as small simulations must be fully explored. All our simulations were carried out on the Cray-YMP and made use of a highly vectorized code written by McDonald [5] which employs a specialized vectorization-compatible selection rule for modelling collisions (see Baganoff and McDonald [6]) and various programming steps taken to improve code efficiency, as discussed by McDonald [5] and Baganoff [7]. The resultant computational speed of the code was roughly 1.0 microsecond per particle per time step.

In defining the problem to be studied, performance considerations led to the selection of simulated wind tunnel dimensions of 40 cells in the streamwise direction, 55 cells in the vertical direction (half space) and 3 cells in depth for a total of 6,600 cubical cells. The vertically oriented flat plate had a half height of 10 cells and a streamwise thickness of 3; see Figs. 1-3. The upstream mean free path length was set at 2.0 cells, giving the Knudsen number of 0.1 quoted above. On varying the average particle number density (based on the entire simulation) from roughly 8 to 121 particles per cell, the overall size of the simulations thus varied from roughly 53,000 to 800,000 particles. The fairly large upper limit was the controlling factor in our selection of a two-dimensional problem for study, as opposed to a three-dimensional problem. Our use of a small 3-unit depth for the simulated wind tunnel, while modelling a two-dimensional problem, was related to the three-dimensional capability of the code used. This particular selection of parameters resulted in a run time of approximately one second per time step for the larger simulations, and a corresponding total run time of roughly 0.5 hrs. The reference solution, see Eq. (1) below, employed 800,000 particles and 1,689 time steps for-time averaging the data.

Statistical Error

In order to determine the level of statistical fluctuations, or *rms* error, associated with a given simulation, one must consider two items: first, a reference solution is needed against which all others are compared; and second, a specific definition for the measure of *rms* error must be introduced. For the rarefied flow considered, an exact solution is simply not available to provide a reference. However, it will be shown that a procedure can be found for determining the absolute *rms* error for each run, from an analysis of the entire group of runs, even without having the exact solution itself among the group. This apparent logical contradiction becomes more rational when one learns that at least one high quality solution must be present in the group to give reliable results; and that the results of the full analysis are not much different from the straightforward approach of using the highest quality run, consisting of the largest number of particles and the longest time averaging, as the reference.

With regard to defining the *rms* level of statistical fluctuations, it is clear that the macroscopic fluid quantities density, velocity, temperature, pressure, stress, and heat flux may exhibit different levels because they represent different moments of the velocity distribution function. Because density is the zeroth-order moment, it should exhibit the smallest ratio of *rms* error to mean, while pressure and temperature represent second-order moments and the corresponding ratios should be higher. Therefore, the analysis must distinguish between the different macroscopic variables. Most of the results given below will be presented for the temperature variable. In defining a single numerical measure of error for a particular macroscopic variable, one could consider a single point in the flow that corresponds to a particular position of interest or consider an average for the entire flow field. The definition to be applied will make use of an average over the flow field.

The appropriate concepts are most easily reviewed if the simplest approach is considered first, i.e., the case consisting of the largest number of particles and the longest time averaging is used as the reference, and all other runs are compared with it. In a simulation, a macroscopic fluid quantity is first determined from an appropriate average of data associated with the particles in a single cell and then it is further averaged as the simulation is advanced in time. Using the overbar notation to designate a time average, the symbol $\bar{q}_{\alpha,i}$ will be used to represent a time-averaged

macroscopic fluid quantity q associated with run α and evaluated for cell i. If the corresponding reference quantity $\bar{q}_{r,i}$ is viewed as an exact mean value, then the square error can be defined by

$$\delta^2_{\alpha,i} = (\bar{q}_{\alpha,i} - \bar{q}_{r,i})^2. \tag{1}$$

A single dimensionless measure of relative error for the entire flow can then be introduced by writing

$$\hat{\mu}^2_\alpha = \frac{1}{N_{cells}} \sum_{i=1}^{N_{cells}} \left(\frac{\bar{q}_{\alpha,i}}{\bar{q}_{r,i}} - 1 \right)^2, \tag{2}$$

where $\hat{\mu}_\alpha$ has the interpretation of a dimensionless rms value. An alternative to (2) that simplifies the computation somewhat makes use of a single reference mean value, such as the maximum, and the corresponding relation reads

$$\mu^2_\alpha = \frac{1}{N_{cells}} \sum_{i=1}^{N_{cells}} \left(\frac{\bar{q}_{\alpha,i}}{\bar{q}_{r,max}} - 1 \right)^2. \tag{3}$$

On comparing the value of a fluid variable at the stagnation point, or a point of maximum, to its free stream value, the ratios for density, temperature, and pressure are roughly 15, 15, 100, respectively, for the case studied; see Figs. 1-3. Because random fluctuations scale with the size of the local mean value, it is clear that the relative error μ_α is heavily weighted by the large values near the plate while $\hat{\mu}_\alpha$ is more evenly weighted. Figures 4 and 5 give the results from a series of tests for $\hat{\mu}_\alpha$ and μ_α, respectively, for the fluid temperature variable. The independent variable in the two figures is the average number of particles per cell defined by $N_c = N_{total}/N_{cells}$ and the parameter that was varied was the size of the sample for the entire run defined by $S = N_{total}T$, where T is the number of time steps used in the time averaging. In this analysis all time steps were used in the averaging, none was skipped. Generally, factors of two were used in varying the quantities N_{total} and T. Comparison of Figs. 4 and 5 shows that the respective curves look very similar except for their absolute numerical values, which are different because of the different normalization used in (2) and (3). The datum point that is missing in the two figures is the one for which the reference would be compared to itself. All the curves show the same trend, namely, that the relative rms error decreased monotonically with increasing N_c for fixed values of the total sample size S. Likewise, it also decreases monotonically with increasing T for fixed values of N_c. Because the computational effort grows in proportion to S,

the data show that for fixed computational cost and for most of the region studied, increasing N_c is clearly more effective than increasing T in producing a small *rms* error. Likewise, expression (3) is preferred over (2) because it is more convenient to evaluate and yet it predicts essentially the same results.

Alternatively, a theoretical determination of the absolute *rms* error, as opposed to the relative *rms* error, can be found by first considering two distinct runs (α, β), each carried out with a different number of particles and/or a different duration of time averaging. If $\bar{q}_{\alpha,i}$ and $\bar{q}_{\beta,i}$ represent time averaged data for the same cell but for two different runs, then one is able to define a measure of their difference by

$$\delta^2_{\alpha\beta,i} = (\bar{q}_{\alpha,i} - \bar{q}_{\beta,i})^2. \tag{4}$$

Now, both $\bar{q}_{\alpha,i}$ and $\bar{q}_{\beta,i}$ can be considered to be composed of the exact mean value plus an error; and therefore, their difference is merely the difference of the two absolute error terms alone. Consequently, we may write

$$\delta^2_{\alpha\beta,i} = (\bar{e}_{\alpha,i} - \bar{e}_{\beta,i})^2. \tag{5}$$

If an average is again taken over all cells in the flow, then (5) leads to the relation

$$\Delta^2_{\alpha,\beta} = \frac{1}{N_{cells}\,\bar{q}^2_{r,max}} \sum_{i=1}^{N_{cells}} \left(\bar{e}^2_{\alpha,i} - \bar{e}_{\alpha,i}\bar{e}_{\beta,i} + \bar{e}^2_{\beta,i}\right), \tag{6}$$

where the constant $\bar{q}^2_{r,max}$ is arbitrarily introduced to nondimensionalize the equation. In view of the definition of the absolute error term $\bar{e}_{\alpha,i}$, it is reasonable to treat it as a random variable with respect to its subscript i; and surely its mean value is zero. In addition, the two quantities $\bar{e}_{\alpha,i}$ and $\bar{e}_{\beta,i}$ ought to be statistically independent since they derive from two independent runs. On this basis the cross product term in (6) is expected to vanish when the sum is carried out over all cells ($N_{cells} = 6,600$ for the example studied), thus reducing the relation to

$$\Delta^2_{\alpha,\beta} = \sigma^2_\alpha + \sigma^2_\beta, \tag{7}$$

where σ^2_α is a dimensionless spatially-averaged absolute error term defined by

$$\sigma^2_\alpha = \frac{1}{N_{cells}\,\bar{q}^2_{r,max}} \sum_{i=1}^{N_{cells}} \bar{e}^2_{\alpha,i}. \tag{8}$$

Equation (7) provides the means for obtaining the absolute *rms* error for each run, even without having the exact solution itself in hand. This follows from the fact that the left-hand side of (7) can be computed directly for the different combination of runs using the spatial average of definition (4), i.e.,

$$\Delta^2_{\alpha,\beta} = \frac{1}{N_{cells}\,\bar{q}^2_{r,max}} \sum_{i=1}^{N_{cells}} (\bar{q}_{\alpha,i} - \bar{q}_{\beta,i})^2. \tag{9}$$

The quantities on the right-hand side of (7) are obtained from the solution of the resultant set of simultaneous equations. Clearly the system is over specified, because there are $r(r-1)/2$ distinct entries in the symmetric matrix $\Delta^2_{\alpha,\beta}$ and only r unknowns σ^2_α, where r is the number of different cases or runs. A discussion of (7) is most easily followed if the runs are first conceptually ordered with respect to their *rms* error, where the smallest is designated as run r. Using this ordering, it is clear that the entries near the diagonal in the symmetric matrix $\Delta^2_{\alpha,\beta}$ are not as useful (small relative error) as those further removed. Therefore, the system of equations can be conveniently reduced, by ignoring the less useful equations, to a properly specified set, without having to resort to a least square error method to solve the entire set. On retaining the subset of (7) for which $\alpha = 1, 2, \ldots (r-1)$ and $\beta = r$ and then arbitrarily including the equation $\alpha = 1$ and $\beta = (r-1)$, a closed set of equations is obtained and it is given by

$$\begin{pmatrix} \Delta^2_{1,r} \\ \Delta^2_{2,r} \\ \cdot \\ \cdot \\ \Delta^2_{r-1,r} \\ \Delta^2_{1,r-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdot & \cdot & 0 & 1 \\ 0 & 1 & \cdot & \cdot & 0 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & 1 & 1 \\ 1 & 0 & \cdot & \cdot & 1 & 0 \end{pmatrix} \begin{pmatrix} \sigma^2_1 \\ \sigma^2_2 \\ \cdot \\ \cdot \\ \sigma^2_{r-1} \\ \sigma^2_r \end{pmatrix}. \tag{10}$$

As expected, the solution of (10) varies slightly with changes in the selection of equations retained and the value of σ^2_r shows the greatest sensitivity to alterations in the selection. Nevertheless, Fig. 6 displays the resulting solution of (10) for our data, again for the temperature variable, and shows that agreement with Figs. 4 and 5 is quite good, demonstrating a firm consistency between the three approaches. The agreement between Figs. 4-6 therefore allows one to conclude that the approach

defined by Eq. (3) is much preferred because of its ease in evaluation while still providing the desired information.

The logical procedure leading to (10) gives spatially-averaged absolute *rms* error values for the flow. Equivalent quantities for individual cells could be found if one were willing to consider a greatly increased computational effort. In Eq. (6) the cross product term vanished because of the spatial averaging. The same term could be made to vanish, while retaining the index i, if an ensemble average were introduced instead. This would allow the same development leading to Eq. (10) except the index i would be preserved, thus giving values associated with individual cells. Clearly the large number of repeated simulations required to carry out ensemble averaging would be prohibitively expensive in practise. However, the concept that such data could be found in principle is important to our understanding of the method.

Computational Cost

The display of data in each of the Figs. 4-6 reflects the order in which the numerical simulations were conducted. For example, consider Fig. 5 and the sequence for which $N_c = 121$ particles/cell. In this case the total number of particles used in the simulation was set, once steady state was reached, and the time averaging was carried out in steps, where the total time-averaging period for each step was double the previous period. If the conditions of a run happen to be compatible with the requirements of the ergodic hypothesis then the *rms* error should decrease as $T^{-1/2}$; and it can be seen from the data for $N_c = 121$ that a factor of 4 increase in T leads to a reduction of *rms* error by a factor of 2, which is consistent with the ergodic hypothesis. However, this rule clearly does not apply for all values of N_c displayed, but it is difficult to judge from the curves where the rule begins to fail. The same observation also applies if the data of Fig. 5 were displayed with T being the independent variable.

On the other hand, if the choice of variables is rearranged as shown in Fig. 7, then the judgement becomes much easier to make. The total sample size $S = N_{total}T$, which also represents the computational cost for the DSMC method, is shown as a function of the average cell particle density N_c, for fixed values of the dimensionless *rms* error μ_α. Because the simulations could not be conducted in this order, these

results were obtained from suitable cross plots of the two graphical forms μ_α versus N_c and μ_α versus T. Focusing attention on the curve for a fixed 4% *rms* error, it is evident that two asymptotes exist. For N_c greater than approximately 100 particles/cell, the ergodic hypothesis clearly applies, i.e., the computational cost is constant and independent of the size of the simulation. This is because in this limit the *rms* error is proportional to $S^{-1/2}$ and therefore a fixed error implies a fixed S; and a fixed S results in a fixed computational cost because it is linearly related to S. Finally, from the definition $S = N_{total}T$, a fixed S allows a free choice of N_{total} (or T) and thus cost is independent of the size of the simulation N_{total}.

Following the same curve for 4% error, we find that for N_c less than approximately 30 particles/cell, the computational cost rises rapidly with just a small decrease in N_c. This is the region in which the simulation becomes very inefficient, because there are too few particles in a cell to adequately model the flow physics. Consequently, one attempts to make up the severe deficiency with a huge increase in the period of time averaging. This asymptotic limit obviously shows that, for a given level of *rms* error, there is a minimum N_c that is allowed, even if the period of time averaging were infinite. In retrospect, this is a conclusion that should be expected on physical grounds; however, the numerical simulations were needed to fix the actual numerical value at which this occurs. The division between efficient and inefficient simulations can be conveniently defined by the knee in the curve, which for the case of 4% error appears at roughly $N_c = 30$ particles/cell.

Continuing to review the curve for 4% error in Fig. 7, we see that a 5-fold increase in N_c from 25 to 125 particles/cell leads to a 10-fold decrease in the computational cost. In other words, a large simulation is less costly than a small simulation! At first glance, this appears to be counter-intuitive, but in actual fact it is merely a reflection of the difference in simulation efficiency at the two extremes. This is a very important conclusion for this class of simulations, because it shows that access to greater computer memory can have a dramatic effect on reducing the computer run time. It also points out that for an extremely large simulation that makes use of all available computer memory and still does not operate in an efficient mode, and which would normally require several hours of run time, sufficient savings in time could be realize by switching to an efficient mode of operation to suggest the possible use of disk read/write to allow the necessary further increase in N_{total}. For this same

4% *rms* error level, we see that roughly 100 particles/cell are required for an efficient simulation and that for the two-dimensional example studied ($N_c = 6,600$) this translates into a total of 660,000 total particles required. A similar three-dimensional problem would require over an order of magnitude greater number (exact ratio obtained from the 55 cell height to 3 cell depth used). Because more than 10 words of data storage are needed for each simulated particle, especially if chemical reactions are modelled, it can be seen that roughly 120 Mwords of memory are needed for an efficient simulation, even when the geometrical resolution of the problem studied is fairly modest, as in our example problem.

Discussion and Conclusions

Many of the past applications of the DSMC method for two- and three-dimensional problems were conducted at average number densities of around 15 to 20 particles per cell. This was done for a number of closely coupled reasons relating to the size of available computer memory, code execution speed, total run time that could be committed, and the type of machine used. The clear conclusion drawn from Fig. 7 is that every effort should be made to employ an average particle number density 4 or 5 times greater, so that full advantage can be taken of the greater simulation efficiency. This is a result that is independent of machine architecture and depends solely on the physics of rarefied gas flow and its simulation. However, the ease with which the desired operating point can be reached is machine dependent and does require appropriate consideration.

The obvious questions left unanswered by this study relate to differences introduced by more complex flow geometries, the presence of multiple species and chemical reactions in the simulated gas, *rms* error specific to a particular cell as opposed to a single measure for an entire simulation, and the effect of varying cell size. The asymptotic limit suggested by each curve in Fig. 7 can be interpreted as the number of particles needed in a single cell to give the same accuracy in a single time step. However, the study was conducted for the case of a steady flow and it does not follow that this same number would necessarily be valid for time accurate results. This question would require a separate study dealing with transient flows. Likewise, in regions of flow where gradients are steep, as occur in regions close to solid boundaries where

translational nonequilibrium becomes very important, one is also not able to conclude from this work that 100 particles per cell is sufficient to give the same 4% resolution, because the boundary layer was relatively thick in the example studied owing to the fairly high Knudsen number chosen. What has been shown is that computational cost for the DSMC method can be reduced in a major way by conducting a simulation in a regime where the relevant physical processes are efficiently modelled, even though the modelling requires the use of significantly greater memory and/or data storage.

References

[1] Alder, B. J. and Wainwright, T. E., "Molecular Dynamics by Electronic Computers," in *Transport Processes in Statistical Mechanics*, I. Prigogine, ed., (Interscience, New York, 1958).

[2] Hockney, R. W. and Eastwood, J. W., *Computer Simulation Using Particles*, (Adam Hilger, Bristol, 1988).

[3] Bird, G. A., *Molecular Gas Dynami* 'Clarendon, Oxford, 1976).

[4] Bird, G. A. , "Monte-Carlo Simulation in an Engineering Context," in *Rarefied Gas Dynamics: Technical Papers from the 12th International Symposium, Progress in Astronautics and Aeronautics Series* (AIAA, New York, 1980) **74**, Part 1, p. 239.

[5] McDonald, J. D. , "A Computationally Efficient Particle Simulation Method Suited to Vector Computer Architectures", Ph. D. thesis, Stanford University, December 1989.

[6] Baganoff, D. and McDonald, J. D. , "A Collision-Selection Rule for a Particle Simulation Method Suited to Vector Computers," Phys. of Fluids A, Vol. 2, No. 7, 1248, (1990).

[7] Baganoff, D. , "Vectorization of a Particle Code used in the Simulation of Rarefied Hypersonic Flow," *Computational Technology for Flight Vehicles*, Vol. I, A. K. Noor and S. L. Venneri, eds., (Pergamon Press, Oxford, 1990). Also in Computing Systems in Engineering, Vol. I, No. 2-4, 437, (1990).

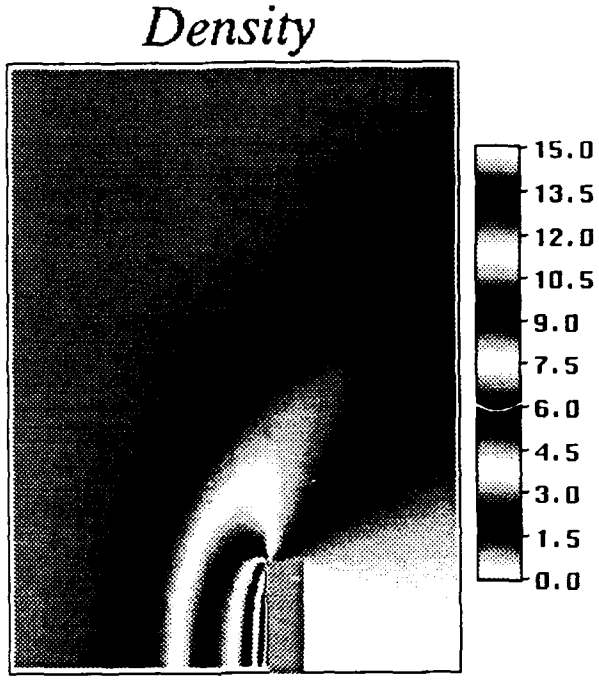

Fig. 1. Density distribution in a rarefied flow about a flat plate for M = 8 and Kn = 0.1.

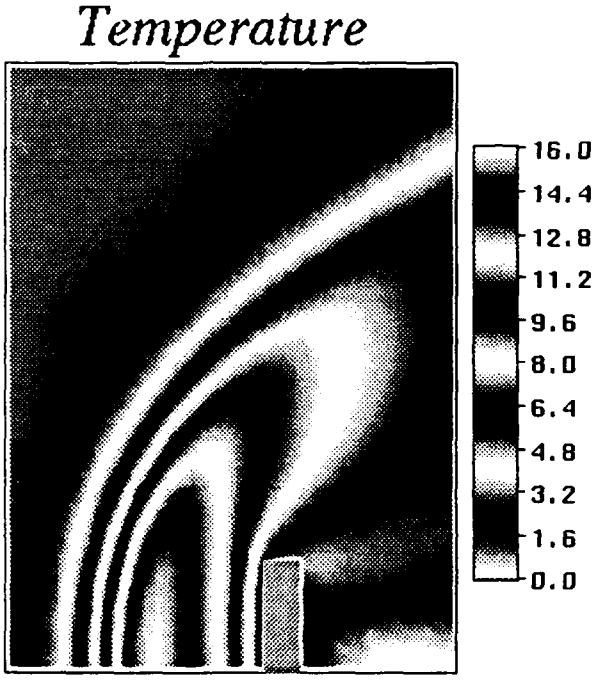

Fig. 2. Temperature distribution in a rarefied flow about a flat plate for M = 8 and Kn = 0.1.

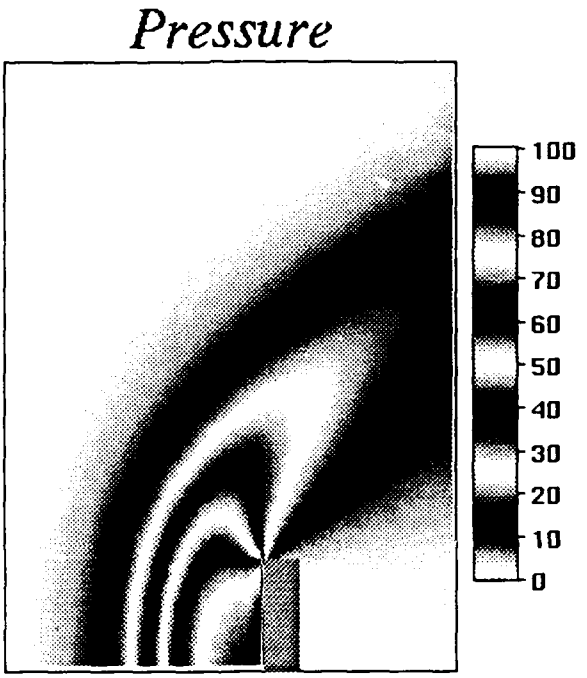

Fig. 3. Pressure distribution in a rarefied flow about a flat plate for M = 8 and Kn = 0.1.

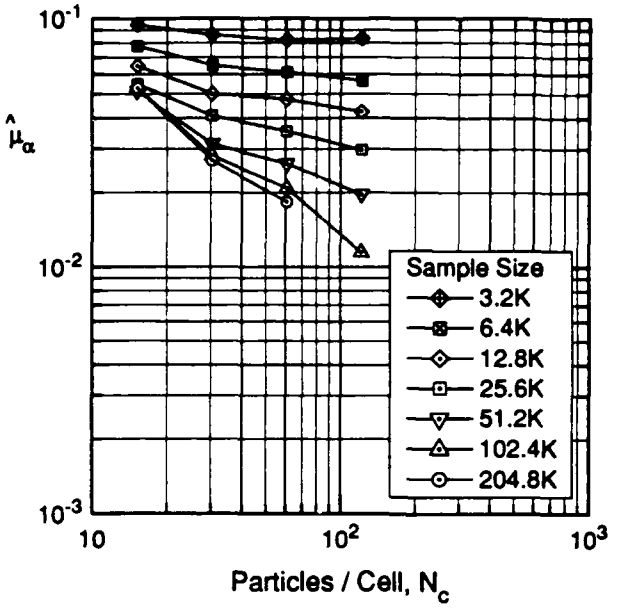

Fig. 4. Relative *rms* error $\hat{\mu}_\alpha$ for the temperature variable versus the average number of particles per cell N_c, holding the total sample size $S = N_{total}T$ fixed.

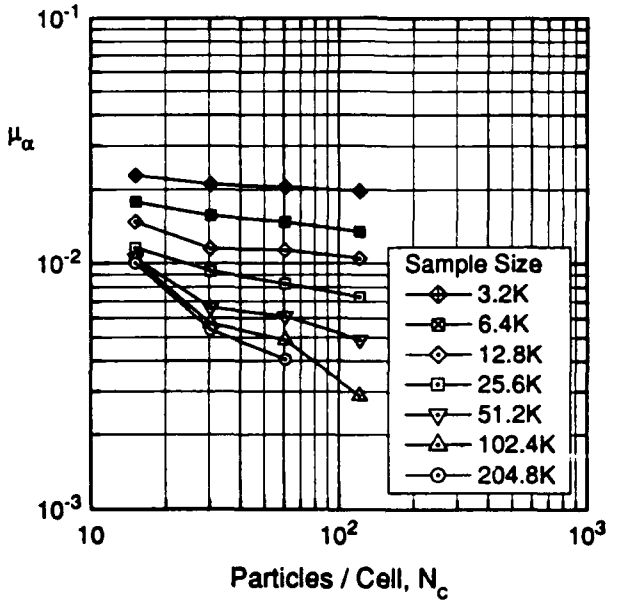

Fig. 5. Relative *rms* error μ_α for the temperature variable versus the average number of particles per cell N_c, holding the total sample size $S = N_{total}T$ fixed.

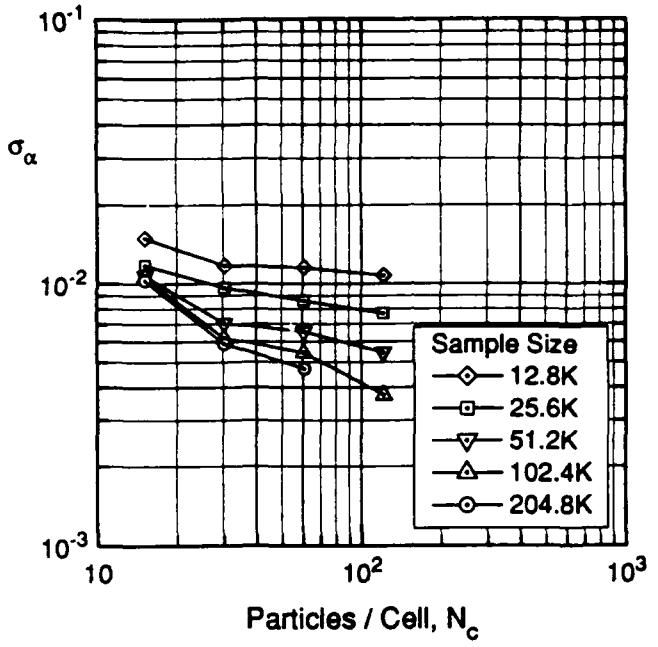

Fig. 6. Absolute *rms* error σ_α for the temperature variable versus the average number of particles per cell N_c, as obtained from the solution of Eq. (10).

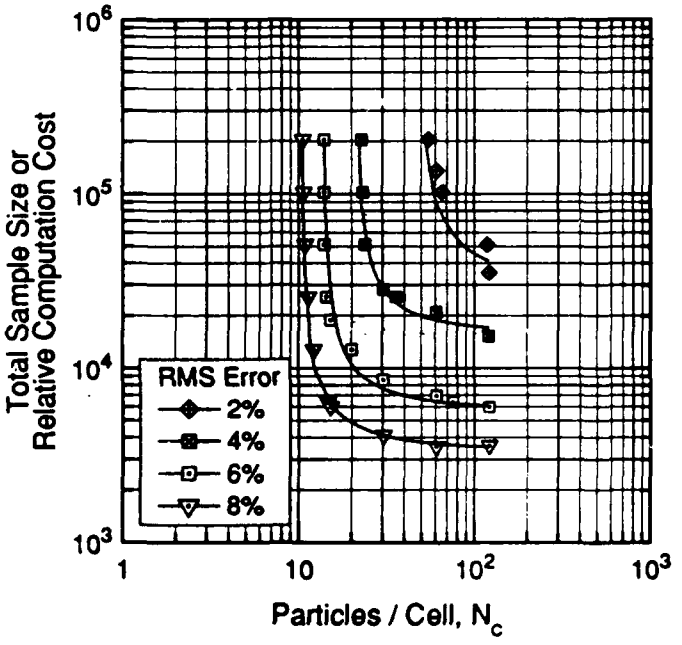

Fig. 7. Total sample size $S = N_{total}T$, or relative computational cost, versus average number of particles per cell N_c for the temperature variable.